The Role of Defense Science and Technology in Software for the Warfighter

By Dr. Charles J. Holland, Deputy Under Secretary of Defense Science and Technology and Mr. Robert Gold, Associate Director for Software and Embedded Systems Office of the Deputy Under Secretary of Defense for Science and Technology

Introduction

Effective technology capability has become the key force multiplier in modern conflict. Defense Science and Technology (S&T) seeks more effective capabilities through better technologies. Accordingly, the mission of Defense S&T is to ensure that warfighters today and tomorrow have superior and affordable technology for revolutionary war-winning capabilities. The results of our S&T fuel the effort to fundamentally transform the way we conduct military operations. Advances in nanoscience and advanced materials; advanced power generation; human dimensions and psychological factors; and directed energy are changing the face of warfighting.

The Defense S&T community has already had countless successes in improving warfighting technology, some recent examples include: stealth technologies; night vision; adaptive optics and lasers; the Global Positioning System; and Phased Array Radars. Some of these technologies successfully migrated to commercial applications. The Internet, formerly the Defense Department ARPANET, is one of the most influential technologies to emerge.

The S&T community continues to drive at the challenges facing our forces today: How do we protect our forces against proliferation of missile technologies, weapons of mass destruction and improvised explosive devices? How do we fight in cities? What type of weapons do we develop? How do we protect our information management systems and infrastructure? We map these problems against the Joint Functional Concepts: Battlespace Awareness; Force Application; Command and Control; Focused Logistics; Force Protection; Joint Operations; Force Management and Net-Centric Operations to ensure our ability to conduct warfare.

To manage the S&T investment — $10 to $11 billion annually, we use the Defense S&T Reliance Process, which is a collaboration between the Director, Defense Research and Engineering (DDR&E) and the Service S&T executives. We also use the process to develop and maintain the Defense S&T Strategy, Basic Research Plan, Defense Technology Area Plan, Joint Warfighting S&T Plan and the Defense Technology Objectives. Our S&T process is influenced by many outside forces. Needs and requirements are validated by the Joint Staff, Congress and DDR&E advisory panels. Our S&T community includes participation by academic institutions, other federal agencies, industry and international partners.

In support of the technical aspects of major defense acquisition, we have institutionalized the Technology Readiness Assessment (TRA) as part of major acquisition reviews. The TRA includes identifying an acquisition program's critical technologies and evaluating those technologies against the NASA Technology Readiness Level scale. Critical technologies with insufficient maturity are identified and a mitigation plan is put into place to ensure that the development efforts mature in time for it to be incorporated into the system. More information on TRAs can be found at http://www.defenselink.mil/ddre/doc/tra_deskbook.pdf.

Lessons Learned

One area of our technology suite that permeates every aspect of defense yet remains to be a challenge is software. The Defense S&T community recognizes that most of our warfighting capability will be enabled by software, so an investment in technologies for managing and developing software is appropriate. Unfortunately, we are still recovering from the view in the late '90s that industry would take care of DoD's software needs.

To highlight some of the ways software has challenged our acquisition programs, we've put together a list of the top six challenges we face in software development today. Some of these challenges are technology related; others rest on the shoulders of program management. Some can be addressed through new tools, techniques and technologies while others require the fortitude to "do the right thing." These challenges are presented as lessons learned so that future programs can avoid these pitfalls and, if successful, return to forums like *CHIPS* to share successes.

1. Believe Your Software Cost And Schedule Estimates

One of the earliest challenges in a program, and often the biggest in terms of far-reaching implications, is having an unrealistic cost and schedule estimate as the basis for the program. Many programs obtain realistic estimates through an independent review or early indications that their estimates are risky. Regrettably, these schedules and inputs are often overridden or ignored by management.

This problem is not isolated to one sector of the military/industrial complex. Both contractor and government managers, under pressure from marketeers, resource sponsors or higher management, succumb to pressure to get the cost and schedule down, which is a good practice. Unfortunately, going too far hurts far worse than it helps because efforts to cut the budget quickly lead to one or more of the other challenges highlighted in this article. The end result is that a 10 percent challenge in cost and schedule may lead to a 200 percent growth rate when a development effort is halted and re-planned partway through because the initial plan was unexecutable.

Calibrated parametric models are reliable, early predictors of cost and schedule for a software project. Resources, both internal and external, are available to provide an independent review. For Major Defense Acquisition Programs (MDAPs), an independent estimate by the Cost Analysis Improvement Group (CAIG) is mandatory.

2. Address System Qualities and Non-Functional Requirements Early

The heady rush to provide new and innovative functionality to the warfighter can cause acquisition and development teams to overlook non-functional requirements and system qualities. Heavy use of commercial-off-the-shelf software (COTS), where these applications cannot be adapted to a program's strategy further complicates the situation, especially during start-up and shut-down scenarios. The ability to understand the internal states of a computer system is probably the most underappreciated item in the development process until problems occur during system integration.

Information Assurance requirements were sometimes hotly debated within major warfighting platforms in the 1980s and early 1990s as to their applicability to embedded tactical systems. Today's thinking readily accepts some measure of IA as an integral part of any IT-enabled system, the question is: *How much?* The lesson here is — don't overlook IA requirements. Incorporate them from the beginning to ensure they are properly addressed.

3. Identify, Find and Solve Technical Problems

Managers are quick to address budget shortfalls by eliminating infrastructure and downsizing development teams as early as possible. But having adequate facilities to support coding, integration and testing is the primary enabler for finding problems that inevitably arise during development. Many of DoD's large acquisition programs nearing completion have suffered as a result of poor decisions in these areas. Lack of opportunities to find problems have had a detrimental impact on acquisition programs' integration and testing efforts. Indeed, the last 20 percent of problems addressed during integration are the integrator's greatest nightmare — the intermittent bug that is difficult to replicate in laboratories. These problems can be incredibly difficult to resolve without the necessary facilities and technically qualified personnel.

4. Avoid Stovepipes

Empowered interdisciplinary teams have been an excellent practice for many years now, but we don't always live up to our best practice in one area critical to software-intensive system developments — integration and test. The trickiest technical problems in software-intensive systems can only be found and solved through effective working relationships between systems engineering, information architects, software developers, ASIC (application-specific integrated circuit) designers and others as needed.

These relationships are often most effective when fostered from the beginning by establishing empowered teams. Another way to state this is: *Bring the software developers out of the closet and accept them into society.* Poor communication between software developers and systems engineers only impedes progress. Integrated teams are an excellent approach to breaking down these barriers.

We have frequently noted that when problems with hardware arise, teams are always quick to break out the models, the analyses, the test results and give the engineer face-time with the managers. When a software problem arises, no one wants to see the smoking code, or delve into the design flaw. The software "glaze" descends over the eyes of managers when software issues are discussed at program reviews. *Please, give software the same energy you would give any other program issue.*

5. Engineer And Test for Off-Nominal and Boundary Conditions

Much like system qualities and non-functional requirements, performance at the edges of the envelope is still overlooked in a complex software development. Our ability to understand and anticipate a performance envelope from a warfighting platform standpoint is pretty good although we still make mistakes. In a computer and information environment, understanding the performance envelope is extremely difficult, especially at the edge of the embedded envelope where the system context can turn seemingly minor computer anomalies into system-critical errors or safety hazards.

While we don't want to detract from engineering for the nominal, our platforms will operate in all areas of the mission space and the software, at critical times, will be expected to perform in less than ideal conditions. The difficulty is that many of these off-nominal conditions can only be tested through simulation. This means that we'll have to take the extra time to ensure our models and simulations are adequately suited for their purpose. Don't scrimp on verification, validation and accreditation (VV&A). Flawed models and simulations can mask critical system errors.

6. Monitor and Manage Critical Resources

Planning and executing any software project is critically dependent on having the right resources; however, traditional software metrics mechanisms fail to address the monitoring of all the critical resources. Most schemes address computer processing resources, data communications resources and staffing. Systems technical resources, such as frame and thread utilization, are often overlooked. Poor use of development facilities can be almost as bad as not having them, so those resources should be monitored as well.

Use of critical facilities (e.g., single board computers, computer-in-the-loop, hardware-in-the-loop) resources often becomes the most chaotic when a program gets into trouble. Moving to parallel build delivery paths to keep progress on multiple fronts, without providing new facilities, merely results in development facility overload, making every build late. In these situations, we recommend restoring order by ensuring activities on these facilities are organized.

This may mean delaying some tasks, but the chaos, if left unchecked, will only ensure that every build will deliver late. Issues with system technical resources, such as exceeding frame or thread processing timelines, can generally be addressed through a technical improvement, if there is time in the schedule to address the change. Not addressing the issue is likely to result in a performance shortfall.

Future Acquisition Challenges

DoD has many exciting agency and Department-level initiatives that carry much promise in transforming our military. The challenge for software and systems folks will be to adapt to the implications of these initiatives. Systems-of-systems (SOS) engineering has arisen to improve interoperability and efficiency. Information and enterprise architectures hold the promise of bringing organizational order to our business IT investments.

They also complement the interoperability objectives of SOS engineering. Evolutionary acquisition allows us to address complexity incrementally, rather than forcing us to produce complex solutions in one sweep. Net-centric operations are driving us to take advantage of the huge amount of data and corresponding opportunities for ad-hoc interdependency once our major systems are networked. All of these emerging trends are somewhat at odds with traditional systems and software project management methods.

Future challenges, as a result of these innovations, will impact software and systems developers at every level. Developers will be expected to perform to higher standards of quality, safety and security based on less-defined and changing requirements while the compile and run-time environments will change more frequently. Evolutionary acquisitions will create a longer-term relationship with the development staff facilitating long-term approaches in product and facility management.

Acquiring Software-Based Functionality

Defense S&T still has modest levels of research investment devoted purely to software technology, a few examples are described below.

Model Based Integration of Embedded Systems (MoBIES)
Model-based software development is an emerging technology for embedded software developers. The Defense Advanced Research Projects Agency's (DARPA) MoBIES program has combined off-the-shelf and research tools in each critical area of software development into an interoperable tool chain that automates most of the mundane tasks associated with software development. Specific development areas addressed by the MoBIES tool suite include: translation of domain needs into design models; translation of design models into run-time models; and translation of run-time models into mathematically sound, near self-evaluating code.

The combination of these tools across an open framework enables tremendous increases in productivity by replacing labor-intensive steps with an almost seamless fabric of tools to automate the delivery of code from systems-engineered needs and ideas. The MoBIES toolset requires domain-specific knowledge to customize these tools to provide specific analysis techniques. Integration technologies facilitate the hardest parts of software development, such as integration from reusable components; automated testing; verification and validation; and auto-generation of optimized runtime implementations on diverse hardware systems.

The MoBIES team also implemented their open tool suite in two different domains: vehicle control and signal processing. MoBIES development tools exceeded expectations by reducing programming staff hours from five days to two minutes to develop signal analysis code for a software programmable radio. Classification of unknown electronics signals into one of three types was conducted more efficiently by the MoBIES-based software as a result of new approaches taken by the systems engineers when working in the MoBIES environment.

Many MoBIES technologies have already transitioned to industry. MoBIES development methods were successfully demonstrated in developing software for vehicle platooning for the automotive industry in San Diego. MoBIES uses advanced technology transition methods by working directly with the OMG (Object Management Group) to document open tool integration standards and is making their developed tools available under a form of open licensing with the Escher Research Institute. Military successes include the F/A-22 and the Army's Future Combat System, with future applicability to the Joint Strike Fighter. The automotive industry has been an equal participant from program inception.

Software Protection Initiative

Through the Air Force Research Laboratory, our office sponsors the Software Protection Initiative (SPI), which is an effort to prevent the unauthorized distribution and exploitation of critical national security application software. Business objectives for this effort are:

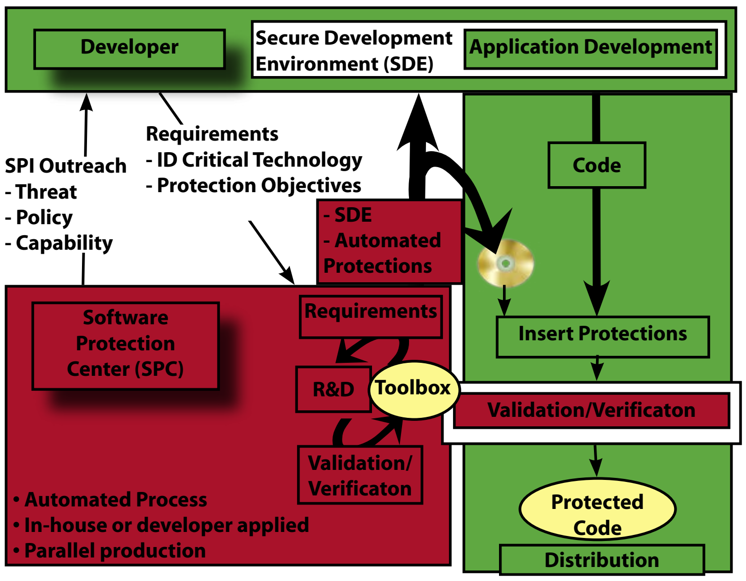
- Inserting protection measures into existing applications
- Measuring the effectiveness of current protection measures
- Researching new protection technology
- Educating the software development community on the software protection philosophy
- Highlighting the threat to high-end software and the need for protection
- Collaborating with the commercial sector on protection methods
- Researching current software protection policy and developing policy

The Air Force Research Laboratory is tasked to manage and execute the SPI with the following goals: (1) deter the acquisition of high-value DoD software by our adversaries; (2) make the exploitation of DoD software cost-prohibitive when it does leak; and (3) ensure that technology and policy protection measures are appropriately applied, balancing mission requirements with security.

SPI efforts have yielded several important technology advances to provide robust and tailored protection for DoD intellectual property in existing applications. These include the development of a Secure Development Environment (SDE) to ensure total life cycle protection in developmental applications and the development of tools to simulate the attack process and accurately measure the level of protection afforded within a given threat environment.

The SPI vision for software development and protection is shown in Figure 1. Typical developer activities are in shown in green. SPI has added the Software Protection Center (SPC), a validated set of tools to empower developers to develop code in a secure environment and apply protections. The toolbox contains a wide array of approved techniques which have a minimal impact on developers, automate the process of protection, and enable parallel implementation. For more information, please contact the AT-SPI Technology Office at (937) 477-3089 or by e-mail: AT-SPI_outreach@wpafb.af.mil.

Figure 1. An illustration of the SPI Vision



High Performance Computing

In 1992, in response to Congressional direction, DoD established the High Performance Computing Modernization Program (HPC-MP) to organize and upgrade the computing infrastructure for our research facilities, test centers and laboratories. Today, the HPCMP vision is to promote a culture for DoD scientists and engineers to use advanced computational environments to solve problems. To that end, the HPCMP provides supercomputer services, high-speed network communications, and computational science expertise that enables Defense laboratories and test centers to conduct a wide range of research, development and test activities.

The HPCMP three components are: HPCMP HPC Centers, Networking and Software Application Support. The HPC centers provide the major computing resources and are further divided into Major Shared Resource Centers and Distributed Centers. The four MSRCs house the bulk of the HPCMP computing resources and provide a full range of computing capabilities including hardware, software, data storage, archiving, visualization, training and expertise in specific computational technology areas. The HPC also includes several Distributed Centers that have modest capabilities compared to the MSRCs but are more widely available to provide convenient access to users who need less capability but easier physical access.

The HPCMP Networking component provides interconnectivity between MSRCs, DCs, and DoD research and test facilities via the Defense Research and Engineering Network (DREN). Researchers and testers can access HPCMP resources remotely via the DREN. Use of the above resources is enabled by infrastructure software applications. HPCMP resources include applications that provide robust modeling, simulation and computation in HPC applications of highest impact to DoD. These products facilitate a large fraction of the DoD S&T, Developmental Test and Evaluation (DT&E) computational workload.

We enhance productivity and capability by providing training, collaboration, tool development, support for software development, technology tracking, technology transfer and outreach to users. For this year, we selected the first High Performance Computing Software Applications Institutes (HSAI) to form a group of experts to accelerate solving DoD's highest priority challenges. By employing cross-Service and multidisciplinary approaches, we hope to make further advances in research and test and evaluation. More information on the HPCMO can be found at http://www.hpcmo.hpc.mil/.

DoD's dependency on software will only grow as we demand more functionality in smaller packaging with lower power consumption. Many of the anticipated warfighting benefits from an increased reliance on networked operations and multisystems engineering will also be realized through software. In spite of our love/hate relationship with software, we still need major investments in technologies to enable us and our industry partners to synthesize the software segments of our systems in a repeatable, manageable, cost effective way resulting in software free from unintentional or malicious defects.

We cannot rely on commercial industry alone to make the advances we need. We must reinvigorate our research programs to provide the tools and techniques by which these software challenges can be met. CHIPS

Dr. Charles J. Holland is the Deputy Under Secretary of Defense Science and Technology. In this capacity, he provides leadership to the entire spectrum of the more than $10 billion annual Defense science and technology portfolio executed through the three Services and Defense Agencies. In addition, he oversees the DoD High Performance Computing Modernization Program, the Defense Modeling and Simulation Office and the Software Engineering Institute, which provide corporate Department of Defense capabilities. His office is also responsible for validating the Technology Readiness Assessment of all major DoD programs requiring Defense Acquisition Board decisions. He is the U.S. Principal to The Technical Cooperation Program (TTCP), a cooperative defense science and technology activity with representatives from Australia, Canada, New Zealand, United Kingdom and the United States.

Over the past two decades, Dr. Holland has played a key role in federal high performance computing R&D. Recently, he served as co-lead for the Report on High Performance Computing for the National Security Community (July 2002) and led the development of the white paper DoD Research and Development Agenda for High Productivity Computing Systems (June 2001), which served as the roadmap for the current DARPA program in high-end computing. He received the Presidential Rank Award, Meritorious Executive (2000), the Society for Industrial and Applied Mathematics Commendation for Public Service Award (1999), and the Meritorious Civilian Service Award from the Secretary of Defense (2001), Air Force (1998) and the Navy (1984). He is a member of the Board of Trustees for the Consortium for Mathematics and its Applications (COMAP) and the Editorial Board of Computing in Science and Engineering. He received Bachelor of Science (1968) and Master of Science (1969) degrees in applied mathematics from the Georgia Institute of Technology and a doctorate (1972) in applied mathematics from Brown University.

Mr. Robert Gold is the Associate Director for Software and Embedded Systems, Office of the Deputy Under Secretary of Defense for Science and Technology. He has 17 years of acquisition experience and has focused on complex software-intensive system development for the last 10 years. Mr. Gold began employment with the Naval Sea Systems Command (NAVSEA) in 1986, where he served in a variety of systems engineering, software engineering and acquisition positions for submarine, surface ship and missile programs.

Mr. Gold is a member of the Professional Acquisition Workforce and is Level 3 certified in both systems engineering and program management. He holds a Bachelor of Science degree in electrical engineering and a Master of Science degree in systems engineering from Virginia Polytechnic Institute and State University.